Elektrotehniški vestnik 77(4): 173-178, 2010
Electrotechnical Review: Ljubljana, Slovenija

Raziskovanje omrežnih napadov: muholovec *Simx*

Mojca Ciglarič [1] **in Simon Mavsar** [2]

[1] *UL, Fakulteta za računalništvo in informatiko, Tržaška 25, Ljubljana*
[2] *Hermes Softlab, Litijska 51, Ljubljana*
E-pošta: mojca.ciglaric@fri.uni-lj.si, simon.mavsar@gmail.com

Povzetek. Z muholovci zajemamo napadalčevo aktivnost za poznejšo forenzično analizo, s čimer omogočajo boljše razumevanje napadalčevih tehnik in orodij ter motivov napada. V članku najprej predstavimo in opišemo koncept muholovca in pregledamo obstoječa orodja. Navajamo njihove glavne pomanjkljivosti in podamo nekaj predlogov, kako bi jih lahko izboljšali. Nato predstavimo prototip lastnega muholovca *simx*, ki vključuje vse predlagane izboljšave: tesnejšo integracijo z operacijskim sistemom, večjo prilagodljivost in uporabnost na širšem naboru sistemov, jasno, pregledno in modularno arhitekturo ter vključitev glavnih tehnik zaznavanja napadov. Po tehničnem opisu prototipa sledi validacija funkcionalnosti prototipa s pomočjo simulacije napada, članek pa se konča s kratko diskusijo prednosti in nadaljnjega razvoja muholovca *simx*.

Ključne besede: muholovec, omrežje muholovcev, vdor, sistem za odkrivanje omrežnih vdorov, omrežna varnost

Network intrusions research: honeypot *simx*

Extended abstract. Honeypot is a tool for capturing the intruder's activity, to be used later in forensic analysis. It enables better understanding of intrusion techniques, tools and motives. The paper first defines the honeypot concept and overviews the existing products. Their disadvantages are discussed and suggestions for better implementations are given. The third section describes *Simx* - our prototype honeypot implementation, which pays due regards to the proposed suggestions. Its architecture is shown in Figure 1 and is described in detail in Section 3.2. Figure 2 shows the architecture of its main component - *Simx* driver. The main mechanisms are explained, especially the intrusion detection techniques and ways of hiding the honeypot functionality from intruders. Listing 1 shows how *Simx* detects port scanning attempts. Section 4 describes a prototype testing environment and results of real and simulated intrusion attempts. Figure 3 shows our test environment and Listing 3 shows sources of the captured unauthorized access attempts. The log of the simulated port scanning attempts is shown in Listing 4 and attack results are logged in Listing 5. Listings 6 and 7 show the logs of the captured activity at the terminal and on the file system, respectively. The last section overviews *Simx* and its main benefits. Although it is currently still in its prototype phase, we believe its future will be bright.

Keywords: honeypot, honeynet, intrusion, intrusion detection system (IDS), network security

1 Uvod

Muholovci so relativno nova in zelo dinamična kategorija varnostnih orodij. Muholovec je sistem, ki ga ne uporabljamo za produkcijske namene, vendar pa želimo zbujati vtis, kot da gre za produkcijski sistem. Na njem bomo beležili vse dostope in aktivnosti. Ker sistem ni produkcijski, bodo vsi dostopi in aktivnosti nelegalni, torej jih imamo lahko za vdor. V nasprotju s požarnimi zidovi ali sistemi za odkrivanje vdorov muholovec brez nadaljnje aktivnosti in analize zajetega materiala ne reši nobenega konkretnega problema. Muholovce je mogoče uporabiti v preventivne namene, za zaznavanje in zajem podatkov. Pojem muholovec opredelimo kot informacijski sistem, katerega glavna vrednost je v njegovi neavtorizirani uporabi [1]. Vse transakcije na tem sistemu so tako neavtorizirane ali škodljive in kot take predmet nadaljnje analize. Primer uporabe je namestitev na spletni ali datotečni strežnik v ločenem segmentu računalniškega omrežja, ki je namenjen izključno raziskovalnim namenom.

Muholovec pa ni nujno le računalniški sistem. Lahko je katera koli digitalna entiteta brez dejanske produkcijske vrednosti: v bolnišnici naredimo lažno elektronsko kartoteko za pacienta z imenom Barack Obama in jo kot muholovsko komponento vstavimo v svojo bazo pacientov. Opazujemo dostope do nje in lete pozneje naprej analiziramo.

1.1 Namen in cilji članka

Članek opisuje novo razvito orodje *Simx*, ki je visoko interaktivni muholovec, primeren za uporabo in nadaljnje raziskovanje. Največkrat takšne muholovce vzdržujejo prostovoljne, neprofitne raziskovalne ali izobraževalne ustanove, ki imajo za cilj zbirati informacije o motivih in taktikah omrežnih napadalcev. Zbrane informacije se lahko potem uporabi za izboljšanje zaščite pred njimi. Raziskovalni modeli muholovcev so zahtevni za postavitev in vzdrževanje, zato jih uporabljajo večje, zlasti raziskovalne, vladne in tudi vojaške organizacije.

Prejet 17. februar, 2010
Odobren 22. april, 2010

Po pregledu obstoječih muholovcev in njihove izvorne kode, kjer je ta le na voljo, ugotavljamo, da bi se jih dalo na številnih mestih izboljšati:

Integracija z gostujočim operacijskim sistemom naj bo implementirana čim tesneje. Tako se zmanjša verjetnost, da bo napadalec odkril nameščenega muholovca.

Orodje naj bo čim bolj prilagodljivo, z manj omejitvami za uporabo ter primerno za čim širši nabor testih in produkcijskih poligonov.

Arhitektura orodja naj bo jasno in modularno zasnovana ter pregledna z namenom olajšati morebitno nadaljevanje razvojnega dela, kajti dobra arhitekturna zasnova je ključnega pomena za uspešen razvoj odprtokodnih projektov.

Orodje naj bo že opremljeno z vsaj osnovnimi tehnikami **zaznavanja poizkusa napada**.

Zato postavimo raziskovalno hipotezo tega članka: mogoče je razviti muholovec, ki bo imel najmanj enako funkcionalnost kot danes najbolj razširjena tovrstna orodja in bo vseboval vse naštete izboljšave. Hipotezo bomo dokazali, če bomo implementirali in uspešno preizkusili prototip takega muholovca.

Danes najbolj razširjen muholovec Sebek [10] je razvit v okviru organizacije Honeypot Project [7, 11] in izvorna koda je prosto dosegljiva. Naš prototip *Simx* povzema znane ideje in koncepte iz muholovca Sebek in drugih, jih izpopolni in združi v zaokroženo celoto ter tako dokazuje, da je mogoče izboljšati parametre delovanja, preglednost kode in učinkovitost sodobnih muholovcev.

V nadaljevanju članka na kratko predstavljamo vdore v računalniške sisteme, koncept muholovca, tehnično opišemo prototip ter uporabljene tehnike in koncepte za implementacijo predlaganih izboljšav. Nato predstavimo uporabo *Simxa* in rezultate analize zajetega prometa na opazovanem poligonu. Članek končamo z izvedbo simuliranega napada, s čimer preverimo celotno delovanje prototipa.

2 Pregled trenutnega stanja

Muholovci so relativno nova tehnologija z velikim potencialom za uporabo na področju računalniške varnosti. Idejni koncept muholovca prvič srečamo v članku Billa Cheswicka [12], ki opisuje daljše obdobje opazovanja napadalca, ki je vdrl v univerzitetni strežnik. Ena prvih aplikacij je opisana v knjigi C. Stolla iz leta 1990 [2], zasnovani na resnični zgodbi, ki se je zgodila avtorju. Na delovnem mestu je odkril računalniški sistem, v katerega je nekdo vdrl, nakar se je avtor odločil, da se bo poizkusil naučiti čim več o vdiralcu in samem načinu vdora. Stoll je na sistemu nastavil lažno okolje, s čimer je zaposlil vdiralca in tako pridobil malo časa. Ideja lažnega okolja je bila slediti vdiralčeve povezave, medtem ko se le-ta ukvarja z nastavljeno pastjo.

Leta 1999 so bile te ideje uporabljene kot osnova za delo skupine Honeypot project [11], katere ustanovitelj in voditelj je Lance Spitzner in je neprofitna raziskovalna organizacija računalniških strokovnjakov, specializiranih za računalniško varnost. V desetletju je skupina objavila veliko teoretičnega in praktičnega dela o muholovcih ter razvila in predstavila tehnike za uspešno postavitev in uporabo računalniških pasti na principu muholovca. Spitzerjeva knjiga "Honeypots, Tracking Hackers" [3] podaja temeljne poglede na koncept in arhitekturo muholovcev ter je kompetenten vir terminologije in definicij obravnavanega področja.

V poznejših letih so nastajali številni predlogi uporabe tovrstnih orodij. Poznamo lepljive muholovce, namenjene za upočasnitev napadalca (Sticky Honeypot), drugi so zasnovani za zmanjševanje neželenega reklamnega prometa [13, 14] ali za zavajanje omrežnih napadalcev [16, 17]. Predstavljeno orodje simx pa spada v kategorijo muholovcev, namenjenih opazovanju, zajemu in analizi napadalčeve aktivnosti [15].

3 Prototip raziskovalnega muholovca *Simx*

Simx je visoko interaktiven raziskovalni muholovec, ki nudi storitve operacijskega sistema brez vmesnih simulacijskih plasti. Ciljna platforma za uporabo so vnaprej rezervirani, ločeni segmenti omrežja, ki ne nudijo nobene realne (produkcijske) storitve in razen raziskovalne nimajo ciljne vrednosti. Tako so vse zajete transakcije, vsi poskusi prijave v sistem ter vsi dostopi do podatkov neavtorizirane narave. Zajete informacije o aktivnostih na sistemu se pošljejo prek prikritega kanala skrbniku v analizo za različne namene: analiza trenutnih trendov, identifikacija novih orodij in metod za vdore, identifikacija napadalcev in njihovih skupnosti, zagotavljanje zgodnjega opozarjanja, predvidevanja in samega razumevanja napadalčevih motivov.

3.1 Opis funkcionalnosti

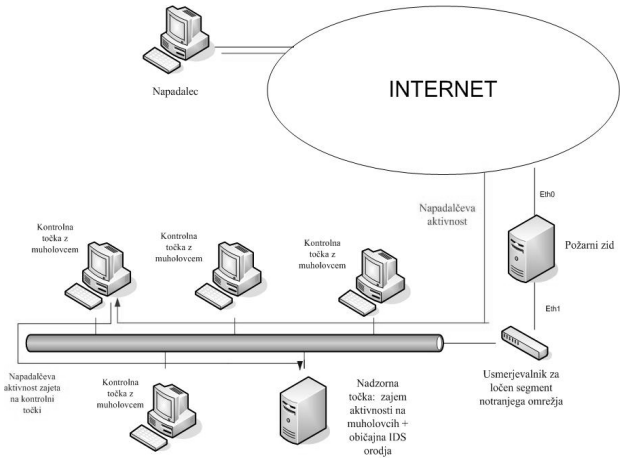
Simx brez napadalčeve vednosti zajame aktivnost na kontrolni točki in le-to na administratorjevo zahtevo pošlje na centralno točko prek prikritega kanala. Primer sistema z nameščenim muholovcem Simx prikazuje slika 1.

Simx omogoča zajemanje več kategorij aktivnosti:

- Aktivnosti na različnih tipih konzol (prestrezanje funkcionalnosti gonilnika tty, kar omogoči zajem aktivnosti na tipkovnici): tty, pts, pty, ptm, cua, console.
- Dostopi do datotek - prestrezanje sistemskih klicev, kar omogoči zajem napadalčeve aktivnosti na datotečnem sistemu: dogodki open (odpiranje), read (branje) in write (pisanje v datoteko).
- Zaznavanje preverjanja odprtih vrat na transportni plasti: SYN, NULL, FIN, ACK, Window, Xmas, Maimon.

Na zahtevo administratorja omrežja se vse informacije o zajeti aktivnosti po skritem kanalu pošljejo na nadzorno točko za nadaljnjo analizo

napadalčeve aktivnosti. Kot transportni protokol lahko skrbnik izbere TCP, UDP ali ICMP.



Slika 1: Arhitektura muholovca *Simx*
Figure 1: *Simx* honeypot architecture

Da ostane *Simx* čim bolj neopažen na računalniškem sistemu in tudi na omrežju, je transportna aktivnost skrita. Zato omrežne pakete (ukazne in podatkovne) označimo z ustreznimi ID polji, ki jih uporabljamo kot kriterij za skrivanje pred ostalimi (napadalčevimi) aplikacijami.

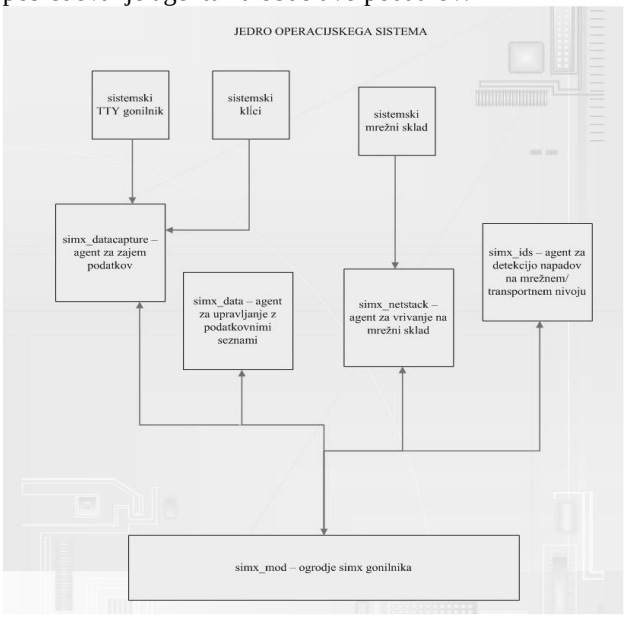
3.2 Arhitekturne značilnosti

Glavni komponenti *Simxa* sta strežnik in gonilnik.

Strežnik *Simx* zbira podatke s kontrolnih točk, na katerih so nameščeni gonilniki S*imx*. Navadno ga namestimo na ločen in zaščiten računalniški sistem v rezerviranem (v raziskovalne namene) segmentu omrežja. Aplikacija je napisana v programskem jeziku C++.

Gonilnik S*imx* namestimo na poljubne kontrolne točke rezerviranega segmenta omrežja. Uporablja se za nadzor napadalčeve aktivnosti na terminalih (uporabljene datoteke ter izhodni omrežni promet), podatke skrbnik na zahtevo zbere na centralni točki za poznejšo analizo. Napisan je v programskem jeziku C kot gonilnik za operacijski sistem GNU/Linux in tako deluje v jedru operacijskega sistema. To omogoča zelo tesno integracijo z gostiteljem, na primer prestrezanje sistemskih klicev, nadzor aktivnosti na omrežnem skladu, manipulacijo omrežnih paketov, skrivanje prisotnosti in aktivnosti … Zajeta aktivnost se interno hrani v pomnilniku do prejema zahteve po določenem tipu podatkov, nakar se zajeti podatki zapakirajo v omrežne pakete in pošljejo po skritem kanalu skrbniku na kontrolno točko. Arhitekturo gonilnika prikazuje slika 2, podrobneje pa so posamezne komponente strežnika in gonilnika *Simx* opisane v preostanku tega razdelka.

Omrežni agent strežnika *Simx* je zasnovan za zajem omrežnega prometa - paketov *Simx*, ki jih pošilja gonilnik *Simx* s kontrolnih točk. Njegova naloga je tudi izluščenje vsebine iz prejetega paketa *Simx* in njeno posredovanje agentu za obdelavo podatkov.



Slika 2: Arhitektura gonilnika *Simx*
Figure 2: *Simx* driver architecture

Agent za obdelavo podatkov strežnika *Simx* bere vhodni tok podatkov, prejet od omrežnega agenta, ga obdela (izlušči vsebino) in predstavi v uporabniku prijazni obliki na standardni izhod ali v dnevniško datoteko.

Podatkovni agent gonilnika shrani zajete podatke v namenskih (glede na tip in izvor podatka) podatkovnih seznamih (dvojno povezani seznami), naslovljivih prek ene glavne liste, ki je ravno tako dvojno povezan seznam. Podatki o zajeti aktivnosti se interno hranijo v pomnilniku, vse dokler administrator ne zahteva prenosa zajete aktivnosti na kontrolno točko.

Agent za zajem podatkov (gonilnik) skrbi za zajem napadalčeve aktivnosti na konzoli, tudi serijski, ter pri oddaljenem dostopu prek telneta / ssh. Spremlja tudi uporabo datotek (dnevnik aktivnosti na datotečnem sistemu) in omrežno aktivnost.

Omrežni agent gonilnika je zasnovan za zajem omrežnega prometa na opazovanem računalniškem sistemu. V ta namen izkorišča možnost direktnega vrivanja v omrežni sklad gostiteljskega sistema.

Agent za odkrivanje vdorov (gonilnik) omogoča zaznavanje različnih tehnik iskanja odprtih vrat na danem računalniškem sistemu. Mogoče ga je širiti, vendar ob upoštevanju dejstva, da jedro operacijskega sistema, v katerem teče gonilnik, ni primerno za kompleksne analize in za iskanje vzorcev v prometu.

3.3 Zajem podatkov o napadalčevi aktivnosti

Gonilnik je zasnovan tako, da omogoča zajem napadalčeve aktivnosti tako na datotečnem sistemu kot tudi na terminalih.

Zajem napadalčeve aktivnosti na datotečnem sistemu je realiziran s prestrezanjem sistemskih klicev, uporabljenih na nivoju VFS datotečnega sistema za vhodno/izhodne operacije nad datotekami.

Zajem napadalčeve aktivnosti na terminalih je realiziran s prestrezanjem funkcijskih klicev različnih tipov gonilnika tty, katerega naloga je upravljanje lokalnih in oddaljenih terminalskih aktivnosti. Gonilnik uporablja tehniko, opisano v [4]: vriva se med tipkovnico in gonilnik tty in izvaja ustrezne operacije glede na generirano aktivnost (torej zajema napadalčevo aktivnost na terminalu).

3.3.1.1 Odkrivanje iskanja odprtih vrat

V jedru lahko z vrivanjem v omrežni sklad operacijskega sistema prestrezamo ves omrežni promet. Simx to počne iz dveh razlogov: za analizo omrežnega prometa in iskanje značilnih vzorcev, ki nastajajo pri iskanju odprtih vrat ter za prikrivanje svojega omrežnega prometa, da ga napadalec ne bi opazil.

Gonilnik *Simx* je trenutno podprt na jedru Linux verzije 2.4.x, kjer imamo za vrivanje na mrežni sklad na voljo koncept netfilter hooks [4], kar omogoča prestrezanje omrežnega prometa na različnih nivojih omrežnega sklada na poti prejetega paketa iz omrežja do uporabniškega konteksta, v katerem tečejo aplikacije.

Ta metoda nam omogoči prestrezanje vhodnega in izhodnega omrežnega prometa. Z analizo zastavic TCP vhodnega prometa lahko tako zaznamo več različnih tehnik iskanja odprtih vrat računalniškega sistema.

Izpis 1 prikazuje izvedbo odkrivanja iskanja odprtih vrat na transportni plasti:

```
uint32_t    sx_ids_check_detect_scan(struct    sk_buff
*a_skb) {
…
        case IPPROTO_TCP: {
            struct tcphdr* tcp_header = \
                (struct  tcphdr*)((char*)a_skb->nh.iph
+ sizeof(struct iphdr));
            // NULL scan
            if (!tcp_header->fin &&
                !tcp_header->syn &&
                !tcp_header->rst &&
                !tcp_header->psh &&
                !tcp_header->ack &&
                !tcp_header->urg &&
                !tcp_header->ece &&
                !tcp_header->cwr) {

                GET_TIME()
                sx_log_dbg(SX_DBG_NETFILTER,
                    "  NULL  scan  detected  from  %s:%d
to port %d",
                    src_ip_addr_str,
                    ntohs(tcp_header->source),
                    ntohs(tcp_header->dest))
             // FIN scan
             else if (tcp_header->fin &&
                !tcp_header->syn &&
                !tcp_header->rst &&
                !tcp_header->psh &&
                !tcp_header->ack &&
                !tcp_header->urg &&
                !tcp_header->ece &&
                !tcp_header->cwr) {

                GET_TIME()
                sx_log_dbg(SX_DBG_NETFILTER,
                    "  FIN scan detected from %s:%d to
port %d",
                    src_ip_addr_str,
                    ntohs(tcp_header->source),
                    ntohs(tcp_header->dest));
                rec.type = pst_FIN;
            // Xmas scan
            else if (tcp_header->fin &&
                tcp_header->psh &&
                tcp_header->urg) {

                GET_TIME()
                sx_log_dbg(SX_DBG_NETFILTER,
                    "  Xmas  scan  detected  from  %s:%d
to port %d",
                    src_ip_addr_str,
                    ntohs(tcp_header->source),
                    ntohs(tcp_header->dest));
```

Izpis 1: Odkrivanje iskanja odprtih vrat
Listing 1: Port scanning detection

3.3.1.2 Kako zagotoviti neopaženost?

Ker je namen muholovca zajem napadalčeve aktivnosti, je treba njegovo prisotnost in aktivnost čim bolj skriti pred napadalcem. V ta namen *Simx* skriva svoj gonilnik in prikriva svoj omrežni promet.

Skrivanje gonilnika: Napadalec po vdoru v računalniški sistem navadno preveri prisotnost orodij, postavljenih za proženje alarmov in pobriše sistemske dnevnike, ki bi razkrili njegovo prisotnost. Ker bi bil tudi gonilnik *Simx* na listi naloženih gonilnikov sumljiv, smo uporabili tehniko [6] vrivanja prevedene gonilniške kode v drugi tip gonilnika, na primer v gonilnik za zvočno kartico (gostujoči gonilnik je poljuben).

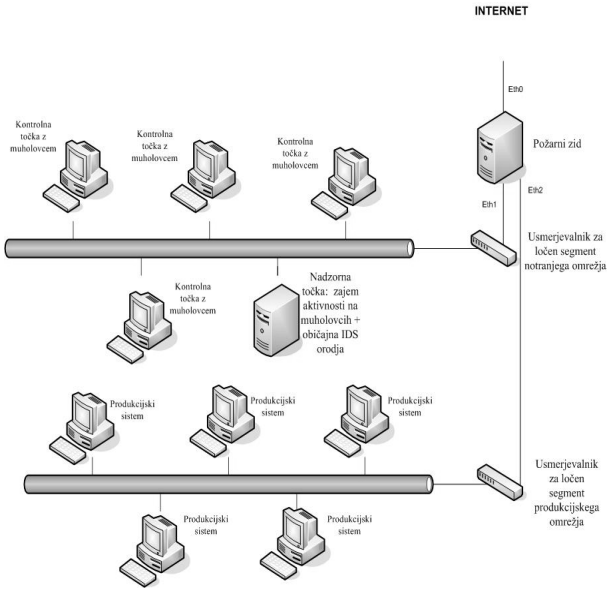
Skrivanje omrežnega prometa, ki ga gonilnik na zahtevo pošilja kontrolni točki: *Simx* za pošiljanje uporabi funkcionalnost omrežne naprave na najnižjem še dosegljivem nivoju omrežnega sklada, da je poslani promet viden na čim manjšem številu nivojev. Pakete pošlje neposredno gonilniku omrežne naprave ter tako zaobide višje nivoje, ki so vidni zunaj jedrnega konteksta, torej tudi omrežnim aplikacijam. Za razlikovanje med omrežnim prometom, generiranim zaradi uporabe muholovca, in preostalim omrežnim prometom, *Simx* označuje »svoje« pakete s pomočjo določenih polj v omrežni in transportni glavi paketa.

4 Uporaba muholovca in rezultati

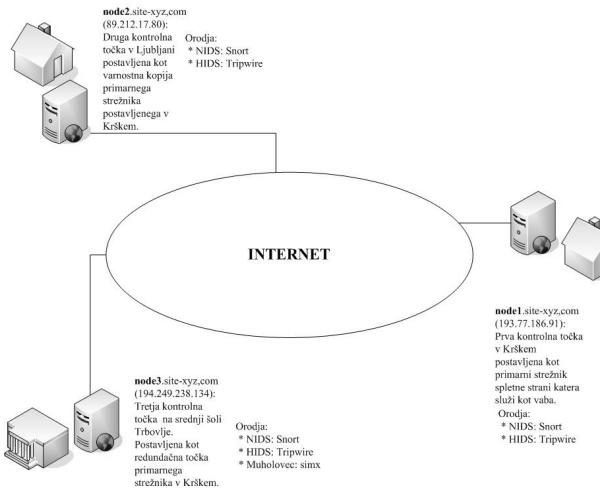
Ciljna platforma oziroma testno okolje (slika 3) za uporabo orodja *Simx* je vnaprej pripravljen segment notranjega omrežja, ki ne ponuja produkcijskih storitev ter nima drugega namena kot raziskovanje. Vse transakcije, vsi poskusi prijave v tak sistem ter vsi dostopi do podatkov na takem sistemu so neavtorizirane narave. Porazdeljen sistem z vabo - spletno stranjo je bil postavljen od aprila do junija 2009 (slika 4).

V nadaljevanju predstavljamo rezultate analize aktivnosti na testnem sistemu. **Najpogostejši tip napada** je bilo izkoriščanje znanih ranljivosti na aplikacijskem nivoju (viewtopic.php, privmsg.php, upload. php) – zloraba pomanjkljivosti v php aplikaciji za pridobivanje dostopa do zaščitenih datotek, izvrševanje

zlonamerne kode ali ukazov SQL. Na odjemalčevi strani so bile prisotne zlonamerne vohunske aplikacije (dogodki tipa SPYWARE-PUT). Sistem je zaznal tudi poskuse pridobitve datoteke z gesli /etc/passwd.



Slika 3: Testno okolje z muholovcem *Simx*
Figure 3: Test environment for *Simx* honeypot



Slika 4: Poligon za zajem podatkov
Figure 4: Test site for data acquisition

Izpis 2 prikazuje število napadov glede na krovno domeno izvora napada za eno izmed opazovanih kontrolnih točk. Za test razvitega muholovca je bila opravljena tudi **simulacija** napada na eno izmed kontrolnih točk. Prvi korak je bila simulacija poizvedovanja – iskanja odprtih vrat z orodjem Nmap [18] z metodo SYN. Izpis 3 prikazuje, kako je *simx* zabeležil to aktivnost. Simulacija direktnega napada je bila izvedena z orodjem Metasploit [8]. Izbrali smo izkoriščanje preplavitve pomnilnika v servisu samba trans2open Overflow [9]. Izpis 4 kaže z orodjem snort zaznani napad – izvršitev zlonamerne kode z nadzorniškimi pravicami, medtem ko nam izpisa 5 in 6 kažeta, kako je *Simx* zabeležil vsebino napadalčeve aktivnosti po napadu – na terminalih in datotečnem sistemu.

```
    %    No      Domain
=======================================
  76.08  684     Network
  60.73  546     US Commercial
  42.27  380     Unresolved
   6.01  54      Slovenia
   2.22  20      Mexico
   2.22  20      Non-Profit
   1.78  16      Seychelles
   1.11  10      Russian Federation
   0.89  8       Switzerland
   0.89  8       Netherlands
   0.89  8       Germany
   0.67  6       United Kingdom
   0.67  6       Czech Republic
   0.67  6       Croatia
   0.44  4       China
   0.44  4       Italy
   0.22  2       Japan
   0.22  2       Trinidad & Tobago
   0.22  2       Australia
   0.22  2       India
```

Izpis 2: Najpogostejši izvori napadov po krovnih domenah
Listing 2: Intrusion root domains by frequency

```
=================================================
 Data dump log:
 Created: 2009/06/14 13:48:05
 Remote IP: 127.0.0.1
 Server version: Simx Server 1.0.25 "PROTOTYPE"
 Drone version: Simx Driver 1.0.25 "PROTOTYPE"
=================================================
[2009/14/06       13:46:47]       portscan       from
192.168.1.13:47754 -> 1410, type (0x4) Syn
[2009/14/06       13:46:47]       portscan       from
192.168.1.13:47754 -> 32777, type (0x4) Syn
[2009/14/06       13:46:47]       portscan       from
192.168.1.13:47754 -> 27374, type (0x4) Syn
[2009/14/06       13:46:47]       portscan       from
192.168.1.13:47754 -> 13783, type (0x4) Syn
[2009/14/06       13:46:47]       portscan       from
192.168.1.13:47754 -> 1447, type (0x4) Syn
```

Izpis 3: Dnevnik zajete aktivnosti iskanja odprtih vrat
Listing 3: Activity log with port scanning acticvity

```
[**] [1:498:6] ATTACK-RESPONSES id check returned root
[**]
[Classification:Potentially Bad Traffic][Priority: 2]
07/18-12:30:55.511182 10.2.2.120:45295 ->
69.44.XXX.XXX:48283
TCP TTL:64 TOS:0x0 ID:56468 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0x2E1F5FBE Ack: 0x47D426D5  Win:
0x16A0 TcpLen:32
TCP Options (3) => NOP NOP TS: 6064395 214022589
```

Izpis 4: Dnevnik rezultata napada
Listing 4: Attack result log

Dnevnik zajete aktivnosti na terminalih omogoča pregled napadalčevih ukazov, izvedenih na sistemu po vdoru. Dnevnik aktivnosti (Izpis 6) na datotečnem sistemu pa omogoča pregled, katere datoteke (in kako) je napadalec uporabil.

```
=================================================
 SIMX Data dump log:
  Remote IP: 193.77.168.91
  Remote hostname: node2.site-xyz.com
  Remote OS: Linux (2.4.78,#2 on i386)
  Drone version: simx drone 1.0.7
  Create time: 2009/05/31 13:46:50
=================================================
```

```
[2009/31/05 13:15:29 0 0] [UP][UP][UP]
[2009/31/05 13:15:38 0 0] id
 [2009/31/05 13:16:14 0 0] [BS]
[2009/31/05 13:16:16 0 0] w
[2009/31/05 13:16:19 0 0] dmesh[BS]g
[2009/31/05 13:17:01 0 0]
[2009/31/05 13:17:01 0 0] unset HISTFILE
[2009/31/05 13:18:01 0 0] lsmod
[2009/31/05 13:19:21 0 0] cat /etc/passwd
```

Izpis 5: Dnevnik zajete aktivnosti na terminalih
Listing 5: Terminal activity log

```
==================================================
SIMX Data dump log:
  Remote IP: 193.77.168.91
  Remote hostname: node2.site-xyz.com
  Remote OS: Linux (2.4.78,#2 on i386)
  Drone version: simx drone 1.0.7
  Create time: 2009/05/31 13:46:50
==================================================
 [2009/31/05 13:15:38 0 0] open: /etc/group
[2009/31/05 13:16:09 0 0] open: /etc/passwd
[2009/31/05 13:16:14 0 0] open: /var/log/wtmp
[2009/31/05 13:16:16 0 0] open: /var/log/lastlog
[2009/31/05 13:16:59 0 0] open: /etc/ld.so.cache
[2009/31/05 13:17:01 0 0] open: /etc/passwd
[2009/31/05 13:17:02 0 0] read: /etc/passwd
```

Izpis 6: Dnevnik zajete aktivnosti na datotečnem sistemu
Listing 6: File system activity log

5 Sklep

Muholovec *Simx* omogoča enako funkcionalnost zajema podatkov o napadalčevi aktivnosti kot danes največ uporabljano orodje Sebek in podobna orodja, vendar z uspešno vključitvijo vseh predlaganih izboljšav: tesna integracija z operacijskim sistemom, velika prilagodljivost, jasna, modularna in pregledna arhitektura, vključitev osnovnih tehnik zaznavanja napadov. Koda gonilnika je skrita v gonilnik za zvočno kartico, zajem napadalčeve aktivnosti na terminalih je izveden z vrivanjem in ne s prestrezanjem sistemskih klicev, kar je bolj prikrito. Prav tako je prikrita omrežna aktivnost. Mogoča je razširitev komunikacije s kontrolno točko pri prenosu zajetega prometa – poleg protokola UDP lahko uporabimo TCP ali ICMP. Preglednost arhitekture je dosežena zaradi razvoja sistema povsem od začetka. Večina ostalih rešitev se je namreč razvijala po načelih odprte kode s prispevanjem več avtorjem in postopnim nadgrajevanjem in dodajanjem funkcionalnosti, kar pa ni omogočalo celovite zasnove.

Nadaljnja prednost *Simx*a je vgrajena komponenta IDS, ki do sedaj ni bila tipičen del muholovca. Jedro operacijskega sistema sicer ni primerno za kompleksne analize ali iskanje vzorcev v omrežnem prometu, kljub temu pa vsaj osnovna analiza prometa lepo spada v muholovec, saj omogoča takojšnje opozarjanje na iskanje odprtih vrat in na napad, ki navadno sledi tovrstnemu skeniranju računalniškega sistema.

Iz vsega naštetega sledi, da smo uspešno implementirali in preizkusili prototip muholovca, izpolnili smo vse zahteve, ki smo si jih zadali, tako pa lahko tudi potrdimo našo začetno raziskovalno hipotezo.

Simx je odprtokodni projekt, ki ga lahko najdemo na spletnem naslovu [19]. Zaradi naštetih prednosti verjamemo, da je pred njim lepa prihodnost.

6 Viri in literatura

[1] The Honeynet Project, Know Your Enemy (2nd Ed.), Addison-Wesley, 2004, ISBN 0-321-16646-9.

[2] Clifford Stoll, The Coocoo's egg: Tracking a Spy Through the Maze of Computer Espionage, The Bodley Head Ltd 1990, ISBN 978-0743411462.

[3] Lance Spitzer, Honeypots: Tracking hackers, Addison-Wesley, 2003. ISBN 978-0321108951.

[4] Phrack, Hacking the Linux Kernel Network Stack, Volume 0x0b, Issue 0x3d. Dostopno na: http://www.phrack.com/issues.html?issue=61&id=13 (2009).

[5] THC-vlogger - an advanced linux kernel based keylogger, http://linux.wareseeker.com/System/thc-vlogger-2.1.1.zip/332763 (2009).

[6] Phrack, Infecting loadable kernel modules, Volume 0x0b, Issue 0x3d. Dostopno na: http://www.phrack.com/issues.html?issue=61&id=10 (2009).

[7] The Honeynet project - A community of organizations actively researching, developing and deploying Honeynets and sharing the lessons learned. Dostopno na: http://www.honeynet.org/ (2009).

[8] The Metasploit Project - The Metasploit Project is an open source computer security project which provides information about security vulnerabilities and aids in penetration testing. http://www.metasploit. com/ (2009).

[9] Samba trans2open Overflow - This exploits the buffer overflow found in Samba versions 2.2.0 to 2.2.8. This particular module is capable of exploiting the bug on x86 Linux and FreeBSD systems. http://www.digitaldefense.net/labs/advisories/DDI-1013.txt (2009).

[10] Sebek - kernel module installed on high-interaction honeypots for the purpose of extensive data collection. http://www.honeynet.org/tools/sebek (2009).

[11] The Honeynet Project, Know Your Enemy: Statistics - Analyzing the past predicting the future, 2001. Dostopno na:http://www.noncombatant.org/trove/honeynet-papers/stats/index.html

[12] Bill Cheswick, An Evening with Berferd In Which a Cracker is Lured, Endured, and Studied. Usenix 1992. Dostopno: http://cheswick.com/ches/papers/ berferd.pdf

[13] Tiny Honeypot. http://www.alpinista. org/thp/ (2009).

[14] N. Krawtez, "Anti-Honeypot Technology" IEEE Security and Privacy. Vol 2, Nb 1, p. 76-78, 2004.

[15] Honeynet Project, Know Your Enemy: GenII Honeynets Easier to deploy, harder to detect, safer to maintain, 2003. Dostopno na: http://project.honeynet.org/papers/.

[16] Deception Tool Kit domača stran, Fred Cohen & Associates. Dostopno: http://www.all.net/dtk/dtk.html (2009).

[17] F. Cohen et al., A Framework for Deception, Tech. Report, 2001, Dostopno na: http://all.net/journal/deception/Framework/Framework.html

[18] Nmap orodje, http://nmap.org (2009).

[19] *Simx* izvorna koda. http://git.site-xyz.com/git/simx

Mojca Ciglarič je doktorirala leta 2003 na UL FRI. Zaposlena je prav tam kot docentka in vodja Laboratorija za računalniške komunikacije. Raziskovalno se ukvarja s področjem varnosti, komunikacij in porazdeljenih sistemov in je avtorica številnih člankov s tega področja.

Simon Mavsar je diplomiral leta 2009 na UL FRI. Zaposlen je pri Hermes Softlabu kot projektni vodja na oddelku za programske rešitve na področju elektronskega bančništva.